Sandia National Laboratories

Exceptional service in the national interest

ME469: Advection Operators: Monotonicity Through "Unwinding" and "Limiters"

Stefan P. Domino[1,2]

[1] Computational Thermal and Fluid Mechanics, Sandia National Laboratories

[2] Institute for Computational and Mathematical Engineering, Stanford

This presentation has been authored by an employee of National Technology & Engineering Solutions of Sandia, LLC under Contract No. DE-NA0003525 with the U.S. Department of Energy (DOE). The employee owns all right, title and interest in and to the presentation and is solely responsible for its contents. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this article or allow others to do so, for United States Government purposes. The DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan https://www.energy.gov/downloads/doe-public-access-plan.

SAND2018-4536 PE

U.S. DEPARTMENT OF ENERGY

NNSA

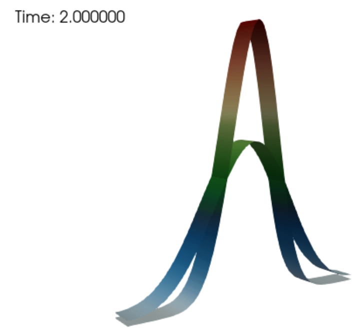
Dissipation and Dispersion Error: Review

Recall, in our modified one-dimensional advection of a passive scalar, we had two types of errors that manifested depending on the underlying numerical approach:

- Dissipative-like error:

$$\frac{\partial \phi}{\partial t} + v\frac{\partial \phi}{\partial x} - \alpha\frac{\partial^2 \phi}{\partial x^2} = 0$$

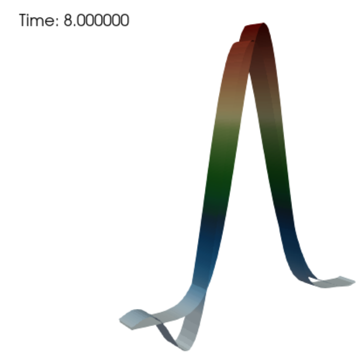
$$\phi(x,t) = e^{(-ivk-\alpha k^2)t}e^{ikx} = e^{ik(x-vt)}e^{-\alpha k^2 t}$$



- Dispersion-like error:

$$\frac{\partial \phi}{\partial t} + v\frac{\partial \phi}{\partial x} + \beta\frac{\partial^3 \phi}{\partial x^3} = 0$$

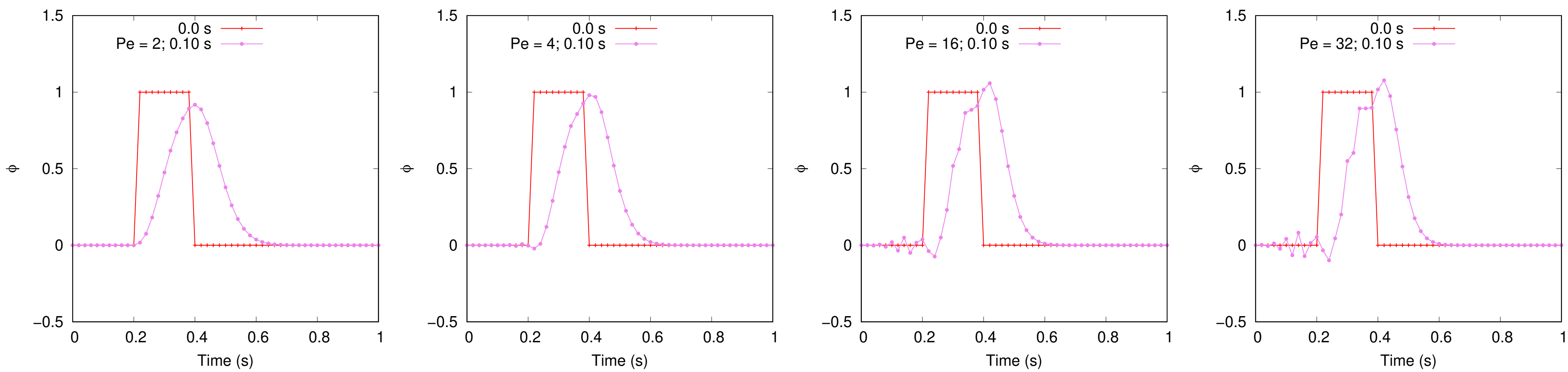
$$\phi(x,t) = e^{(-ivk+\beta ik^3)t}e^{ikx} = e^{ik[x-(v-\beta k^2)t]} = e^{ik(x-wt)}$$



Transient Advection/Diffusion: Step Function as Initial Condition Central, AKA Galerkin

Goal: Run our model equation with a variety of Peclet numbers using a step function as the initial condition

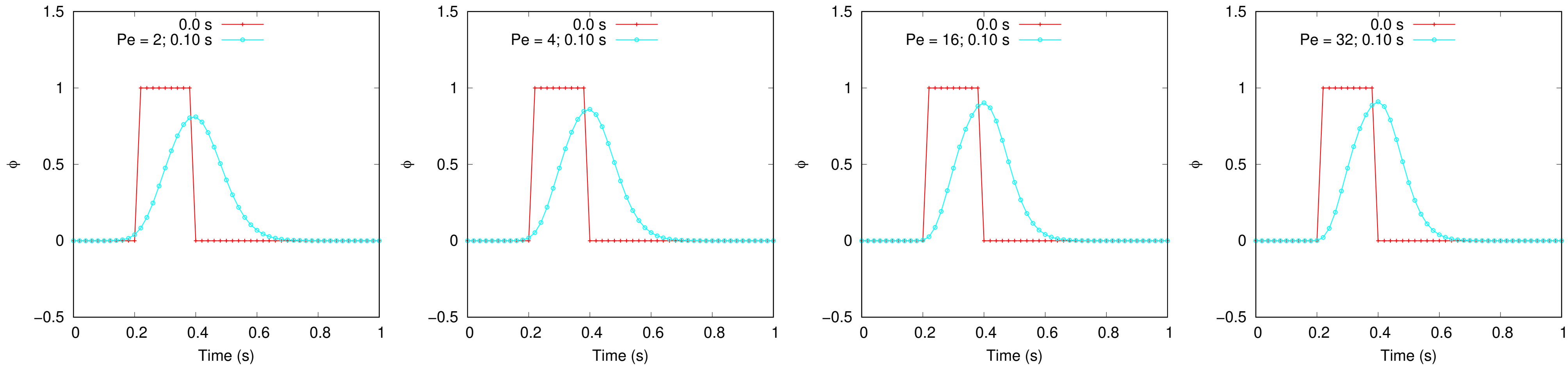
$$\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV \approx \sum_{ip} (\rho u_j)_{ip} \phi_{ip} n_j dS \approx \sum_{ip} \dot{m}_{ip} \phi_{ip} \qquad \phi_{ip}^{CDS} = \sum_n N_n^{ip} \phi_n$$



Transient Advection/Diffusion: Step Function as Initial Condition First-order Upwinf

Goal: Run our model equation with a variety of Peclet numbers using a step function as the initial condition

$$\dot{m}_{ip}\phi_{ip}^{UPW} = \frac{\dot{m} + |\dot{m}|}{2}\phi_L + \frac{\dot{m} - |\dot{m}|}{2}\phi_R$$



Hybrid-Based Blending

For our general temporal advection/diffusion/source equation, can we define an improved, automatic blended approach between central and upwind?

$$\frac{\partial \rho \phi}{\partial t} + \frac{\partial \rho u_j \phi}{\partial x_j} - \frac{\partial}{\partial x_j}\left(\frac{\mu}{Sc}\frac{\partial \phi}{\partial x_j}\right) = S^{\phi}$$

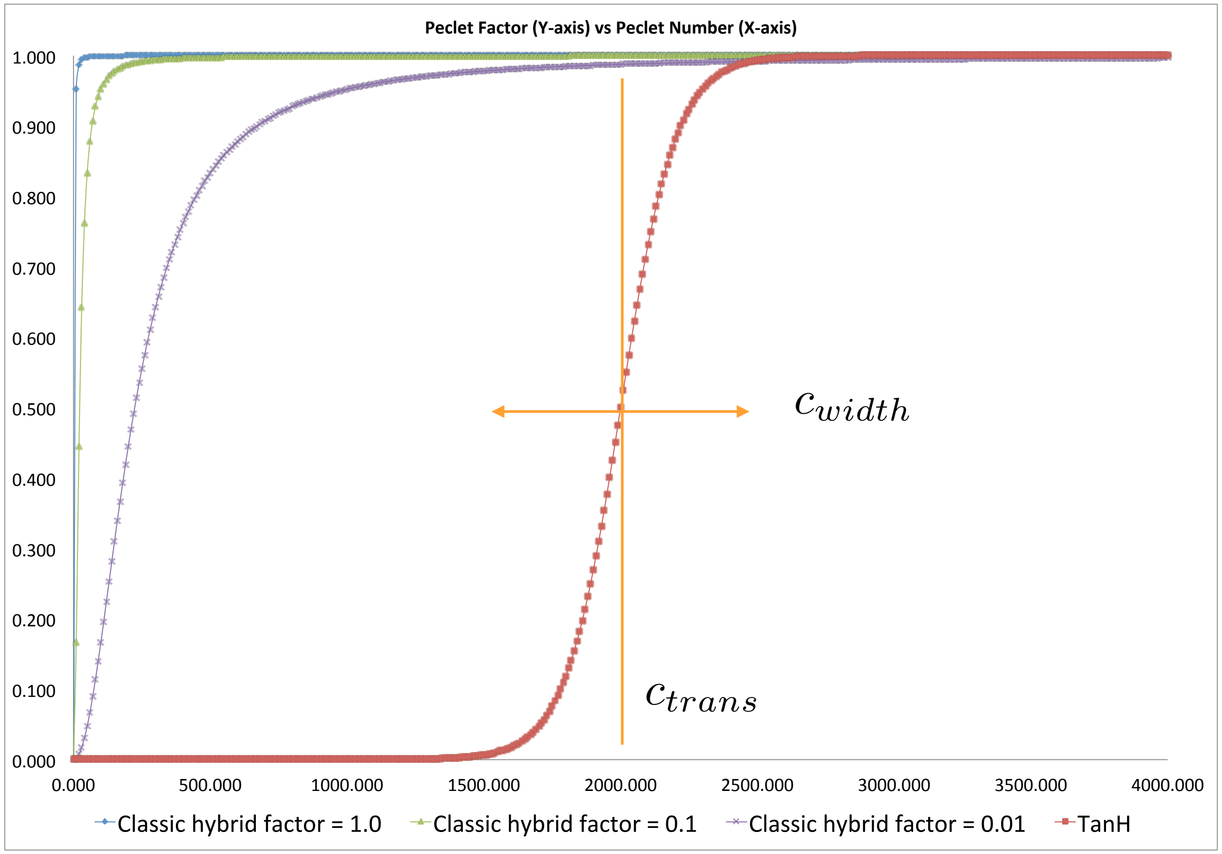
Recall, the advection operator was: $\int w \frac{\partial \rho u_j \phi}{\partial x_j} dV \approx \sum_{ip} \left(\rho u_j\right)_{ip} \phi_{ip} n_j dS \approx \sum_{ip} \dot{m}_{ip}\phi_{ip}$

With (for central, or Galerkin): $\phi_{ip}^{CDS} = \sum_{n} N_n^{ip}\phi_n$

And (for first-order upwind): $\phi_{ip}^{FOU} = \frac{\dot{m} + |\dot{m}|}{2}\phi_L + |\frac{\dot{m} - |\dot{m}|}{2}|\phi_R$

We can define a blended operator as well: $\phi_{ip} = \eta\phi^{FOU} + \left(1-\eta\right)\phi^{CDS}$

Functional form for η – Linked to Peclet number, Pe
Many ad-hoc choices, however, a common physical approach is tanh



$$Pe = \frac{\rho U L}{\mu}$$

$$\eta = \frac{1}{2}\left[1 + tanh\left(\frac{Pe - c_{trans}}{c_{width}}\right)\right]$$

- peclet_function_form:
  - velocity: tanh
  - mixture_fraction: tanh

- peclet_function_tanh_transition:
  - velocity: 5000.0
  - mixture_fraction: 2.0

- peclet_function_tanh_width:
  - velocity: 200.0
  - mixture_fraction: 4.0

Hybrid-Blending Sanity Check

Consider a simple fluids case where we have air (298.15K) flowing 1 m/s in a 1 m^3 domain

- For Pe = 2 at each element, we would require ~0.03 m resolution, or a mesh of size: ~35,000

Consider a simple fluids case where we have air (298.15K) flowing 10 m/s in a 1 m^3 domain

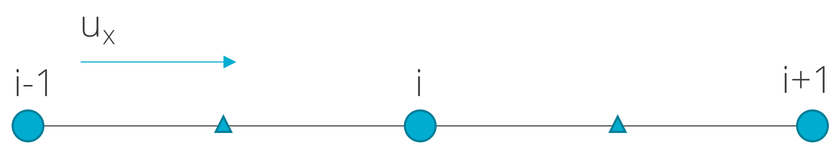
- For Pe = 2 at each element, we would require ~0.003 m resolution, or a mesh of size: ~35,000,000

- In general, this constraint results in extremely high mesh counts for most all practical flow configurations; for turbulent regimes, we would quickly revert to η = unity
- Moreover, as presented, we are blending with upwind – an operator that we have already shown to be overly diffuse and non-energy conserving

Monotonic Issues at High Pe: Simple Matrix Analysis

Consider our passive scalar concentration whose natural range (as a mass fraction) is bounded between zero and unity, here, shown as a stationary transport equation:

$$\frac{\partial \rho u_j \phi}{\partial x_j} - \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial \phi}{\partial x_j}\right) = 0$$



Using our CDS and standard diffusion operator yields the following matrix system:

$$\left(\frac{\rho u}{2}\begin{bmatrix}-1 & 0 & 1\end{bmatrix} + \frac{\rho D}{\Delta x}\begin{bmatrix}-1 & 2 & -1\end{bmatrix}\right)\begin{bmatrix}\phi_{i-1} \\ \phi_i \\ \phi_{i+1}\end{bmatrix}.$$

$$a_{i,i-1} = \frac{\rho D}{\Delta x}\left(1 + \frac{Pe}{2}\right)$$

With coefficients, $a_{i,i}\phi_i = a_{i,i-1}\phi_{i-1} + a_{i,i+1}\phi_{i+1}$ and: $a_{i,i} = (a_{i,i-1} + a_{i,i+1})$

$$a_{i,i+1} = \frac{\rho D}{\Delta x}\left(1 - \frac{Pe}{2}\right)$$

Substituting: $a_{i,i-1} = a_{i,i} - a_{i,i+1}$ and defining: $\xi = \frac{a_{i,i+1}}{a_{i,i}}$

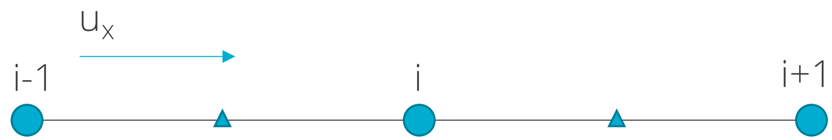
Yields: $\phi_i = \xi\phi_{i+1} + (1-\xi)\phi_{i-1}$

Positive for Pe < 2

For Pe < 2, the value of the scalar at node i is a linear combination of the neighboring values

Monotonic Issues Resolved When Using First-order Upwind

$$\frac{\partial \rho u_j \phi}{\partial x_j} - \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial \phi}{\partial x_j}\right) = 0$$



$$a_{i,i-1} = \frac{\rho D}{\Delta x}(1 + Pe)$$

Positive for all Pe

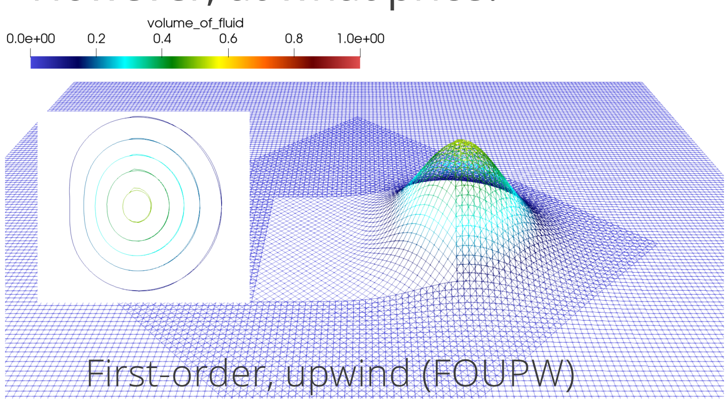
$$a_{i,i} = (a_{i,i-1} + a_{i,i+1})$$

$$a_{i,i+1} = \frac{\rho D}{\Delta x}$$

$$\left(\frac{\rho u}{2}\begin{bmatrix}-1 & 1 & 0\end{bmatrix} + \frac{\rho D}{\Delta x}\begin{bmatrix}-1 & 2 & -1\end{bmatrix}\right)\begin{bmatrix}\phi_{i-1}\\ \phi_i \\ \phi_{i+1}\end{bmatrix}$$

$$\phi_i = \xi\phi_{i+1} + (1-\xi)\phi_{i-1} \qquad \xi = \frac{a_{i,i+1}}{a_{i,i}}$$

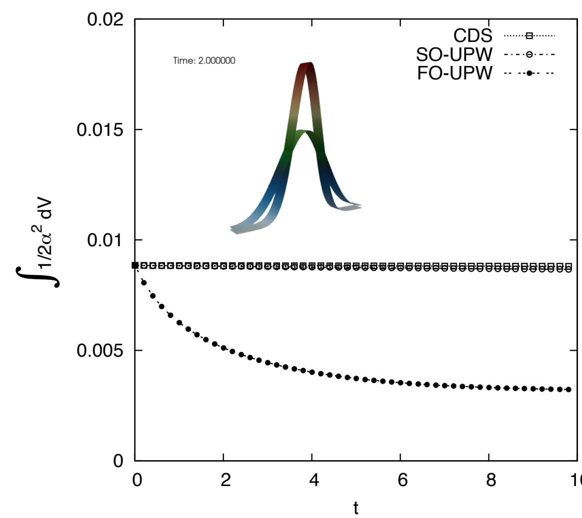
However, at what price?



First-order, upwind (FOUPW)

Fully bounded with diagonal dominance

$$\frac{\sum_{i \neq j} |a_{i,j}|}{|a_{i,i}|} \leq 1$$



Volume-of-fluid example (Domino and Horne, Renew. Ener. 2022)

Monotonic Issues at high Pe: Alternative View (Diagonal Dominance)

Define Diagonal Dominance as: $$\frac{\sum_{i\neq j}|a_{i,j}|}{|a_{i,i}|} \leq 1$$

For a monotonic operator and ease of solving the linear system, diagonal dominance is desired

In our advection/diffusion case, this is expressed as:

$$\frac{\left|\frac{\rho D}{\Delta x}\left(1-\frac{Pe}{2}\right)\right| + \left|\frac{\rho D}{\Delta x}\left(1+\frac{Pe}{2}\right)\right|}{\frac{2\rho D}{\Delta x}} \leq 1$$

Which, again, is only ensured when the Peclet number is less than two

Alternatives to First-order Upwind: Higher-order Upwind

The remaining set of slides are algebra-intensive!

- We do not intend for you to memorize these upcoming formulas, only the philosophy by which they are derived

Alternatives to First-order Upwind: Higher-order Upwind

Recall, that in the finite difference context, we could increase the upwind stencil to increase accuracy, e.g.,

Derivative	Accuracy	−8	−7	−6	−5	−4	−3	−2	−1	0
1	1								−1	1
	2							1/2	−2	3/2
	3						−1/3	3/2	−3	11/6

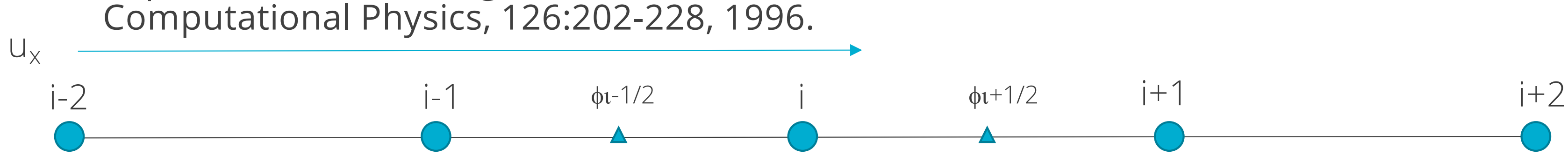
- For fluids: Varying reconstruction approaches:

Essentially Non-oscillatory (ENO)

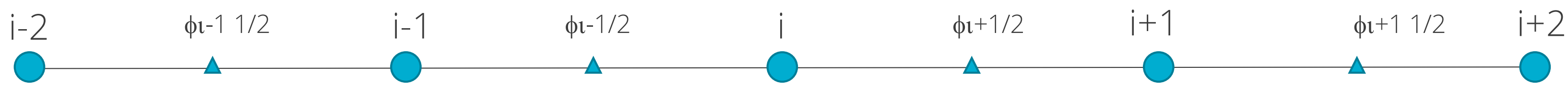
- A. Harten, B. Engquist, S. Osher and S. Chakravarthy, Uniformly high order essentially non-oscillatory schemes, III, Journal of Computational Physics, 71:231-303, 1987.

Weighted-ENO (WENO)

- 3rd-order, Liu, X., Osher, S. and Chan, T. (1994) Weighted Essentially Non-Oscillatory Schemes. Journal of Computational Physics, 115, 200-212. https://doi.org/10.1006/jcph.1994.1187
- Arbitrary-order, G. Jiang and C.-W. Shu, Efficient implementation of weighted ENO schemes, Journal of Computational Physics, 126:202-228, 1996.



ENO and WENO Concept



Define a suite of re-constructions that each provide 3rd-order accuracy:

$$\phi^{(1)}_{i+\frac{1}{2}} = \frac{1}{3}\phi_{i-2} - \frac{7}{6}\phi_{i-1} + \frac{11}{6}\phi_i \quad \phi^{(2)}_{i+\frac{1}{2}} = -\frac{1}{6}\phi_{i-1} + \frac{5}{6}\phi_i + \frac{1}{3}\phi_{i+1} \quad \phi^{(3)}_{i+\frac{1}{2}} = \frac{1}{3}\phi_i + \frac{5}{6}\phi_{i+1} - \frac{1}{6}\phi_{i+2}$$

- The above are simply defined by each of the three-point stencils, e.g.,
  - (i-2, i-1, i), (i-1, i, i+1) and (i, i+1, i+2)

We can also define a fifth-order scheme:

$$\phi_{i+\frac{1}{2}} = \frac{1}{30}\phi_{i-2} - \frac{13}{60}\phi_{i-1} + \frac{47}{60}\phi_i + \frac{9}{20}\phi_{i+1} - \frac{1}{20}\phi_{i+2}$$

- This stencil can be reconstructed based on *linear weights*, γ:

$$\gamma_1 = \frac{1}{10}, \gamma_2 = \frac{3}{5}, \gamma_3 = \frac{3}{10}$$

$$\phi_{i+\frac{1}{2}} = \gamma_1\phi^{(1)}_{i+\frac{1}{2}} + \gamma_2\phi^{(2)}_{i+\frac{1}{2}} + \gamma_3\phi^{(3)}_{i+\frac{1}{2}}$$

Or a convex-combination of weights: (sum to unity):

$$\phi_{i+\frac{1}{2}} = w_1\phi^{(1)}_{i+\frac{1}{2}} + w_2\phi^{(2)}_{i+\frac{1}{2}} + w_3\phi^{(3)}_{i+\frac{1}{2}}$$

The weights, $w_{1,2,3}$, are a function of a “smoothness” factor ($\beta_{1,2,3}$) that are dynamically selected

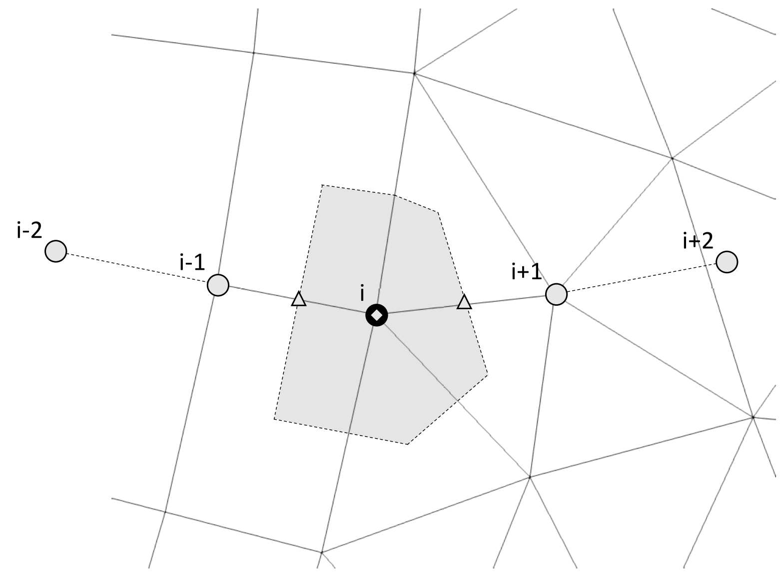
Alternatives to First-order Upwind: Higher-order Upwind – But how on an unstructured mesh?

For an unstructured setting, the former approach is not easily achieved due to the reduced stencil connectivity

- Recall, we are looping edges (as shown), or elements

- How to proceed with reconstructing data that is outside the immediate connectivity?

- Ideas?



Alternatives to First-order Upwind: Higher-order Upwind – But how on an unstructured mesh?

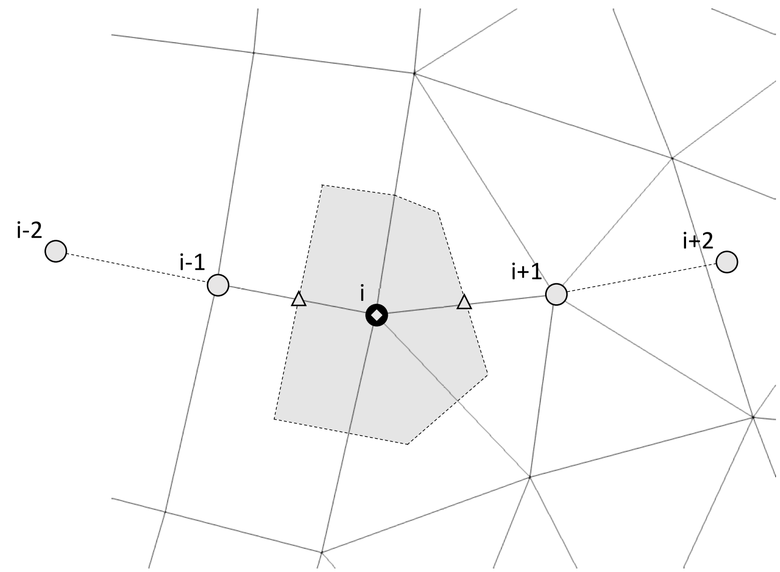
For an unstructured setting, the former approach is not easily achieved due to the reduced stencil connectivity

- Recall, we are looping edges (as shown), or elements

- How to proceed with reconstructing data that is outside the immediate connectivity?

- Ideas?

- Hint… Can we look towards the projected nodal gradient to effectively increase the stencil?



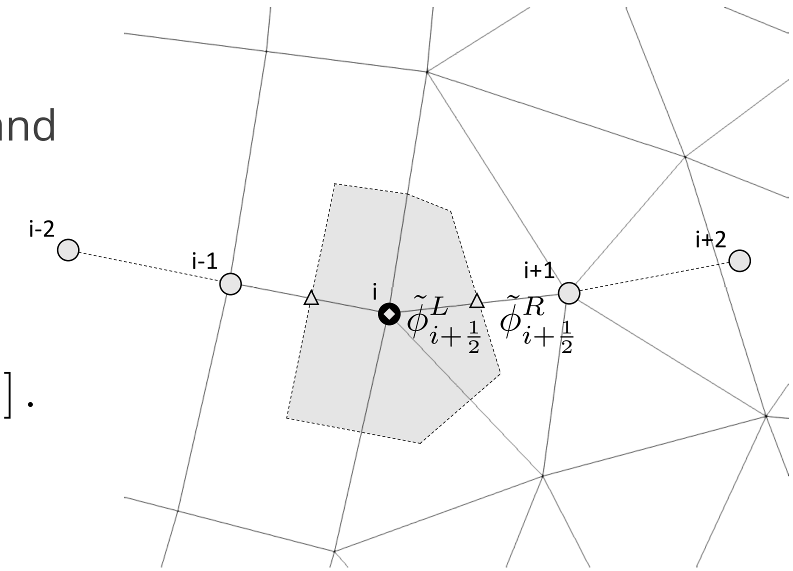
General Kappa-Method of Hirsh

Numerical Computation of Internal and External Flows, vol. 2, John Wiley & Sons, 1990

For the edge defined by the i and i+1 node, define a Left and Right state:

$$\tilde{\phi}^L_{i+\frac{1}{2}} = \phi_i + \frac{1}{4}\left[(1-\kappa)\left(\phi_i - \phi_{i-1}\right) + (1+\kappa)\left(\phi_{i+1} - \phi_i\right)\right],$$

$$\tilde{\phi}^R_{i+\frac{1}{2}} = \phi_{i+1} - \frac{1}{4}\left[(1+\kappa)\left(\phi_{i+1} - \phi_i\right) + (1-\kappa)\left(\phi_{i+2} - \phi_{i+1}\right)\right].$$



- For κ = +1, we simply revert to CDS
- For κ = -1, Second-order upwind
- For κ = 2/3, QUICK (Leonard, “A stable and accurate convective modelling procedure based on quadratic up-stream interpolation”, Comput. Methods Appl. Mech. Eng. 19 (1979) 59–98.)

Assignment: Algebra!!!

Kappa = 0 Method of Hirsh

Numerical Computation of Internal and External Flows, vol. 2, John Wiley & Sons, 1990.

- For κ = 0, recast as: (Algebra.....)

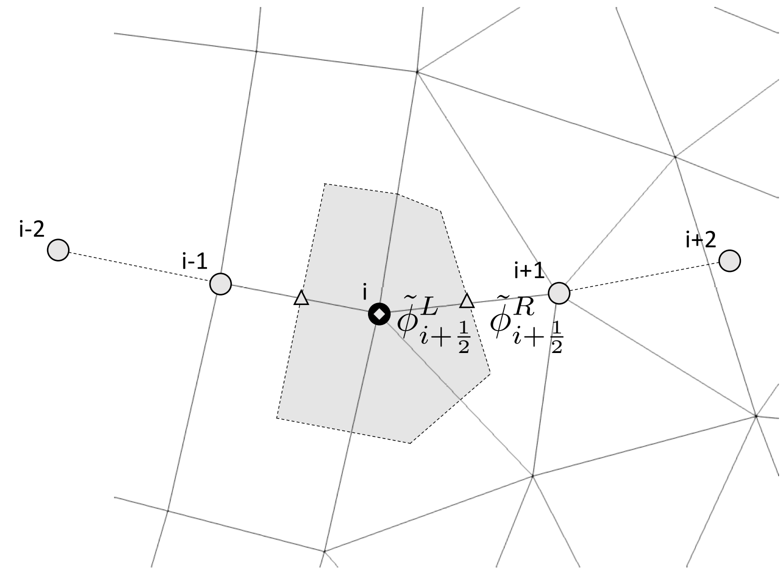
$$\tilde{\phi}^{L}_{i+\frac{1}{2}} = \phi_i + \Phi^L \Delta x^L_j G_j \phi_i,$$

$$\tilde{\phi}^{R}_{i+\frac{1}{2}} = \phi_{i+1} - \Phi^R \Delta x^R_j G_j \phi^{i+1}$$

Where,

$$\Delta x^L_j = x^{ip}_j - x^L_j,$$

$$\Delta x^R_j = x^R_j - x^{ip}_j$$



- Above, define a "limiter" function Φ^L, Φ^R that "senses" when the solution is smooth (tends towards unity) and when the solution is oscillatory (tends towards zero)
- G_j is the projected nodal gradient at each node (or cell-center) that is treated in a *deferred-correction* context, i.e., this quantity is lagged from the previous iteration
- So-called "gradient reconstruction" schemes
  - Reconstruct a higher-order stencil through extrapolation

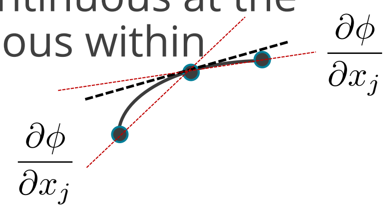
Derived by substituting κ = 0, and using the projected nodal gradient definition – or – just by noting an extrapolation using a gradient

Assignment: Algebra!!!

Projected Nodal Gradient: Refresher

- Objective: We desire a nodal variable that represents the gradient of a scalar ϕ, $G_j\phi$

- We can view the nodal gradient as continuous at the nodes/DOF location, while discontinuous within element/control volume boundaries:



Let's minimize this difference (L_2):

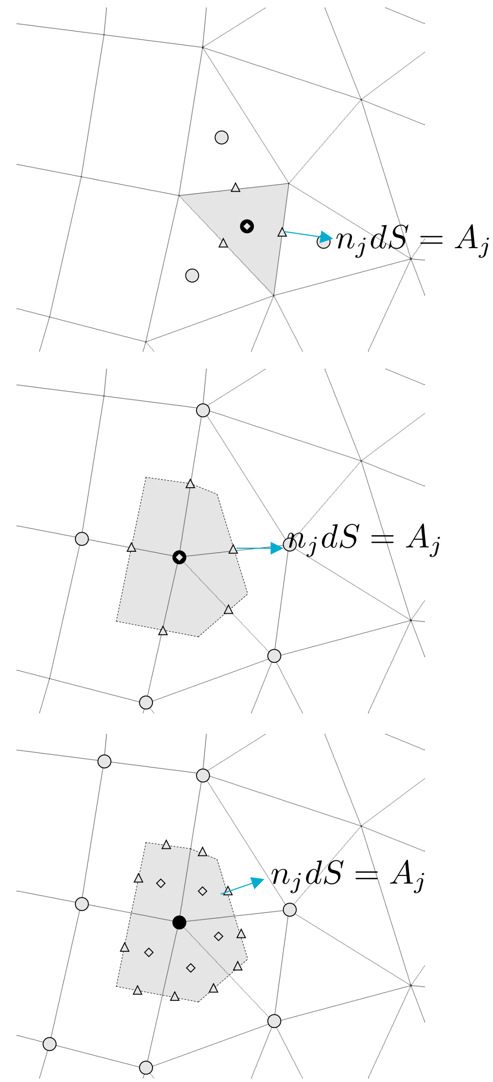
$$\int_\Omega \frac{1}{2}\left(\frac{\partial \phi}{\partial x_j} - G_j\phi\right)^2 d\Omega$$

by solving:

$$\int_\Omega w G_j \phi d\Omega = \int_\Gamma \phi n_j d\Gamma - \int_\Omega \frac{\partial w}{\partial x_j}\phi d\Omega$$

$$G_j\phi = \frac{\sum_{ip} \phi_{ip} n_j dS}{V}$$

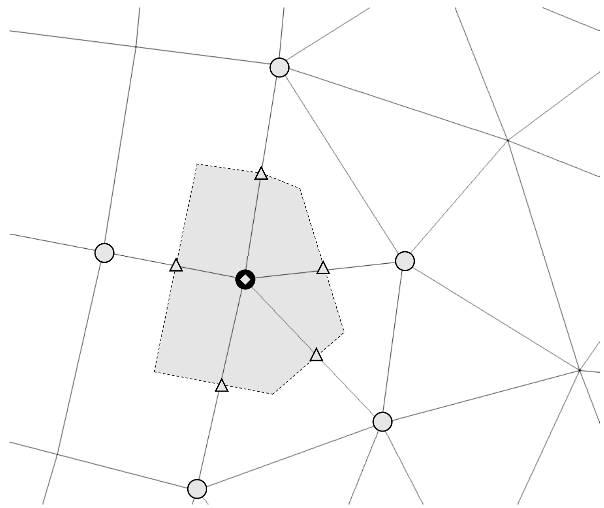
Lumped projected nodal gradient
(piecewise constant w)



Projected Nodal Gradient: Psuedo Code (Edge-based)

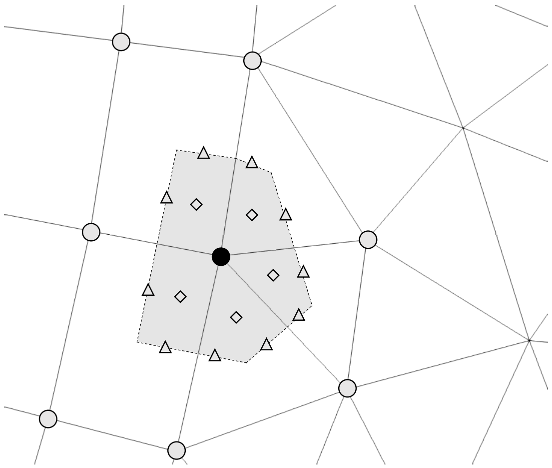
```
for ( stk::mesh::Bucket::size_type k = 0 ; k < length ; ++k ) {
     stk::mesh::Entity nodeL/nodeR = edge_node_rels[0]/edge_node_rels[1];
     const double qL = *stk::mesh::field_data( *scalarQ_, nodeL);
     const double qR = *stk::mesh::field_data( *scalarQ_, nodeR);
     const double qip = 0.5*(qL + qR);
     const double invVolL = 1.0/volL;
     const double invVolR = 1.0/volR;

     for ( int j = 0; j < nDim; ++j ) {
       const double aj = areaVector[k*nDim];
       gradQL[j] += aj*qip *invVolL;
       gradQR[j] -= aj*qip*invVolR;
     }
   }
 }
```

Rule: Area vector points from low to high global node ID

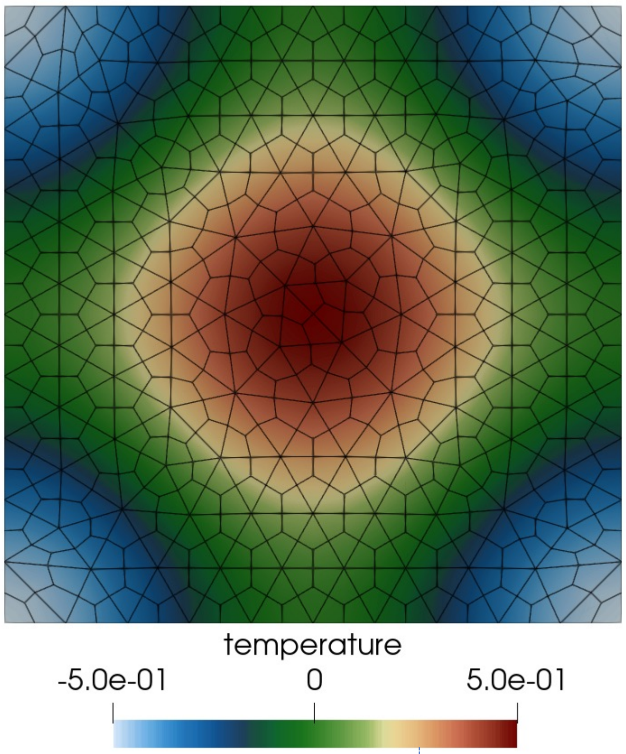


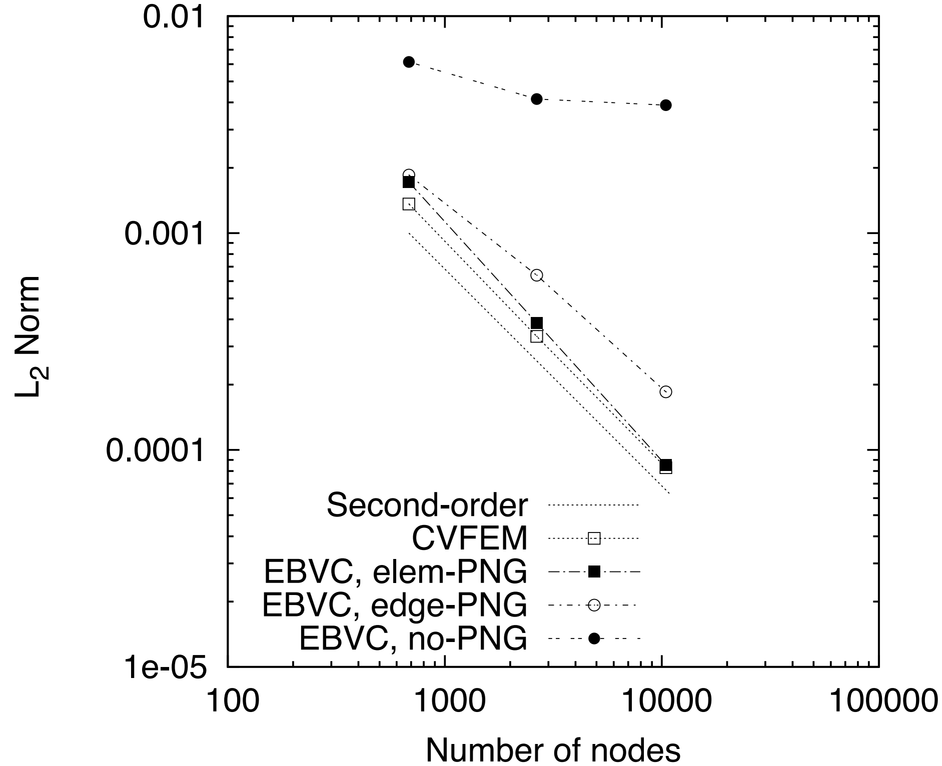
Projected Nodal Gradient: Psuedo Code (CVFEM)

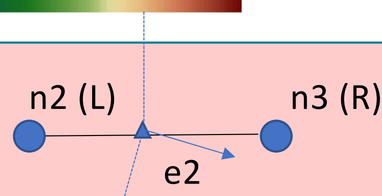
```
for ( int ip = 0; ip < numScsIp_; ++ip ) {

 // left and right nodes for this ip
 const int il = lrscv[2*ip];
 const int ir = lrscv[2*ip+1];

 double qIp = 0.0;
 const int offSet = ip*nodesPerElem_;
 for (int ic=0; ic < nodesPerElem;++ic ) {
   qIp += N[offSet+ic]*p_scalarQ[ic];
 }

 // assemble to il/ir
 for ( int j = 0; j < nDim_; ++j ) {
   double fac = qIp*areaVec[ip*nDim_+j];
   gradQL[j] += fac*inv_volL;
   gradQR[j] -= fac*inv_volR;
 }
}
```



For Instance, Verification of The Diffusion Operator



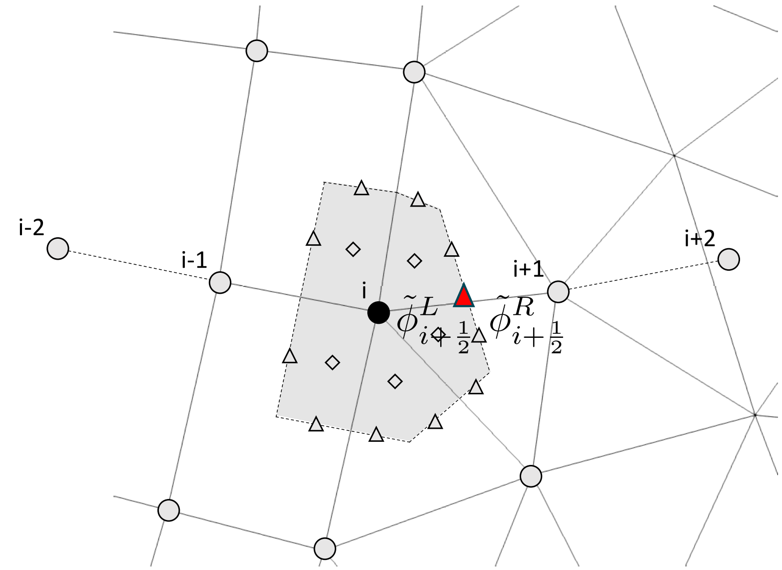




$$\frac{\partial \phi}{\partial x_j}\Big|ip = G_j^{ip}\phi + \left[(\phi_R - \phi_L) - G_l^{ip}\phi\Delta x_l\right]\frac{A_j^{ip}}{A_k\Delta x_k}$$

Kappa = 0 Method of Hirsh: CVFEM

Numerical Computation of Internal and External Flows, vol. 2, John Wiley & Sons, 1990.



- For κ = 0, recast as: (Algebra.....)

$$\tilde{\phi}^L_{i+\frac{1}{2}} = \phi_i + \Phi^L \Delta x^L_j G_j \phi_i,$$

$$\tilde{\phi}^R_{i+\frac{1}{2}} = \phi_{i+1} - \Phi^R \Delta x^R_j G_j \phi^{i+1}$$

Where,

$$\Delta x^L_j = x^{ip}_j - x^L_j,$$

$$\Delta x^R_j = x^R_j - x^{ip}_j$$

- Above, define a "limiter" function Φ^L, Φ^R that "senses" when the solution is smooth (tends towards unity) and when the solution is oscillatory (tends towards zero)
- G_j is the projected nodal gradient at each node (or cell-center) that is treated in a *deferred-correction* context, i.e., this quantity is lagged from the previous iteration
- So-called "gradient reconstruction" schemes
  - Reconstruct a higher-order stencil through extrapolation

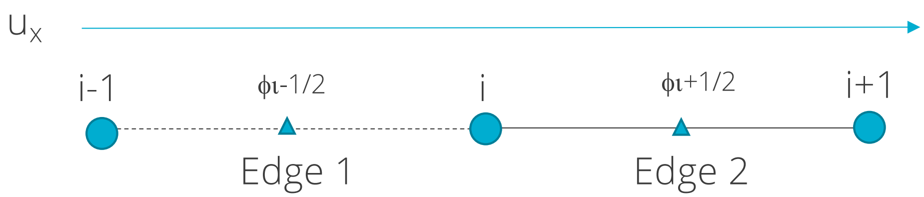
CVFEM surface integration point can be along the edge, or at the standard SCS location with modified distance verctir

Assignment: Algebra!!!

Classic Flux Limiters

Van Leer, B. (1974), "Towards the ultimate conservative difference scheme II. Monotonicity and conservation combined in a second order scheme", J. Comput. Phys., 14 (4): 361–370

- Consider our standard three-point stencil obtained by iterating edge 1 and 2



- In the above stencil, we are stressing that when iterating edge 2, we do not have information easily obtain for edge 1 (shown above as a dashed line); (speaking from an unstructured perspective)

$$\phi_{i+\frac{1}{2}} = \phi^{LOW}_{i+\frac{1}{2}} - \Phi(r_{i+\frac{1}{2}})\left(\phi^{LOW}_{i+\frac{1}{2}} - \phi^{HIGH}_{i+\frac{1}{2}}\right)$$

Monotonic upstream-centered scheme for conservation laws (MUSCL)

- Above, the “LOW” and ”HIGH” are any operators that you select, e.g.,

$$\phi^{LOW}_{i+\frac{1}{2}} = \phi_i \qquad \phi^{HIGH}_{i+\frac{1}{2}} = \frac{\phi_i + \phi_{i+1}}{2} \longrightarrow \phi_{i+\frac{1}{2}} = \phi_i + \frac{1}{2}\Phi(r_{i+\frac{1}{2}})\left(\phi_{i+1} - \phi_i\right)$$

$$r_{i+\frac{1}{2}} = \frac{(\phi_i - \phi_{i-1})}{(\phi_{i+1} - \phi_i)} \qquad (\phi_i - \phi_{i-1}) = 2G_x\phi_i\Delta x - (\phi_{i+1} - \phi_i) \qquad G_x\phi_i = \frac{\phi_{i+1} - \phi_{i-1}}{2\Delta x}$$

Flux Limiter Definition

Sweby (1984) defined set of permissible limiter regions for the desired second-order accurate methods: A sampling of the Sweby Diagram

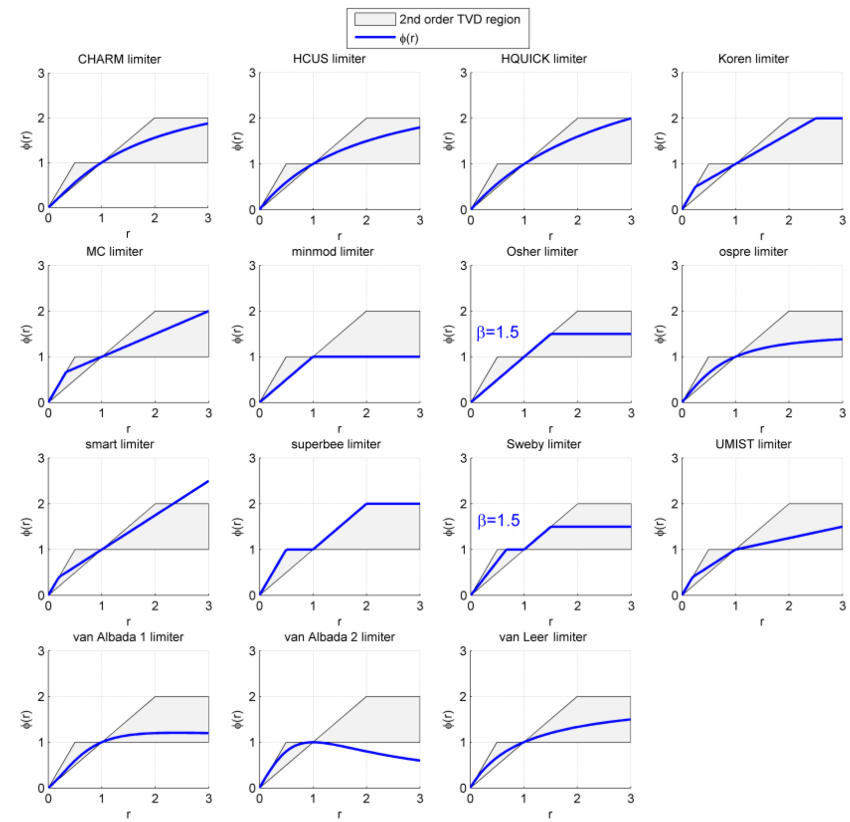
Superbee: $\Phi(r) = max[0, min[2r, 1], min[r/2]]$

Van Leer: $\Phi(r) = \dfrac{r + |r|}{1 + |r|}$

Symmetry property: $\Phi(1/r) = \dfrac{\Phi(r)}{r}$

Total Variation: $TV(\phi) = \sum_{j=1}^{N} |\phi_j - \phi_{j-1}|$

- For a monotonically increasing function, TV(ϕ) = $|\phi_1 - \phi_N|$. Note that if ϕ_1 and ϕ_N are taken constant, then, as long as the function remains monotonic the total variation is constant
- However, if TV(ϕ) increases, then this suggest oscillations in the solution have occurred



Sweby's 2nd order TVD region. Created in Matlab. Griffgruff 18:37, 11 October 2006 (UTC)

Blending Approaches to Arrive at Pseudo Higher-order Methods: Upwind and Central

Define an upwind operator now at an arbitrary integration point:

$$\begin{aligned}\phi_{ip}^{UPW} &= \alpha_{upw}\tilde{\phi}_{ip}^{L} + (1-\alpha_{upw})\,\phi_{ip}^{CDS}; \dot{m} > 0,\\ &\quad \alpha_{upw}\tilde{\phi}_{ip}^{R} + (1-\alpha_{upw})\,\phi_{ip}^{CDS}; \dot{m} < 0.\end{aligned}$$

And for a generalized CDS scheme:

$$\phi_{ip}^{GCDS} = \frac{1}{2}\left(\hat{\phi}_{ip}^{L} + \hat{\phi}_{ip}^{R}\right), \qquad \begin{aligned}\hat{\phi}_{ip}^{L} &= \alpha\tilde{\phi}_{ip}^{L} + (1-\alpha)\,\phi_{ip}^{CDS},\\ \hat{\phi}_{ip}^{R} &= \alpha\tilde{\phi}_{ip}^{R} + (1-\alpha)\,\phi_{ip}^{CDS}\end{aligned}$$

- Two new blending parameters: α_{upw} and α

The Idealized Stencil Set

With Nalu input file specifications

$i-2$	$i-1$	i	$i+1$	$i+2$	α	α_{upw}
0	$-\frac{1}{2}$	0	$+\frac{1}{2}$	0	0	n/a
$+\frac{1}{8}$	$-\frac{6}{8}$	0	$+\frac{6}{8}$	$-\frac{1}{8}$	$\frac{1}{2}$	n/a
$+\frac{1}{12}$	$-\frac{8}{12}$	0	$+\frac{8}{12}$	$-\frac{1}{12}$	$\frac{2}{3}$	n/a
$+\frac{1}{4}$	$-\frac{5}{4}$	$+\frac{3}{4}$	$+\frac{1}{4}$	0	$\dot{m} > 0$	1
0	$-\frac{1}{4}$	$-\frac{3}{4}$	$+\frac{5}{4}$	$-\frac{1}{4}$	$\dot{m} < 0$	1
$+\frac{1}{6}$	$-\frac{6}{6}$	$+\frac{3}{6}$	$+\frac{2}{6}$	0	$\dot{m} > 0$	$\frac{1}{2}$
0	$-\frac{2}{6}$	$-\frac{3}{6}$	$+\frac{6}{6}$	$-\frac{1}{6}$	$\dot{m} < 0$	$\frac{1}{2}$

- alpha_upw:
  velocity: 1.0

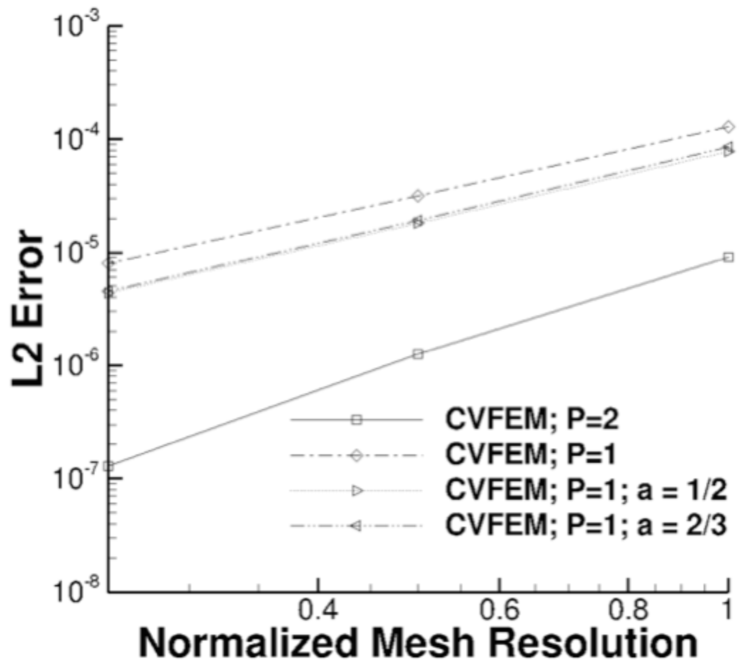
- alpha:
  velocity: 1.0

- upw_factor:
  velocity: 1.0 (zero reverts to first-order)

- limiter:
  velocity: [yes/no]

Pseudo 4th order Verification Results

Verification using Central (linear and quadratic) compared to pseudo 4th order



Lower error, however, formally second-order accurate